LOW LEVEL DESIGN (LLD)

Credit Card Default Prediction

Written By:- REHAN ALVI

Document Control:

Change Record:

Version	Date	Author	Comments
0.1	25-march-2024	Rehan Alvi	Introduction & Architecture defined
0.2	27-march-2024	Rehan Alvi	Architecture & Architecture Description appended and updated
0.3	28-march-2024	Rehan Alvi	Unit Test Cases defined and appended

Reviews:

Version	Date	Reviewer	Comments
1.0	5-April-2024	Rehan Alvi	Document Content , Version Control and Unit Test Cases to be added

Approval Status:

Version	Date	Reviewed By	Approved By	Comments
1.1	10-April-2023	Rehan Alvi		Final

Contents

o. K Nearest Neighbors Classification

p. Boosting

q. Model Validation

r. Flask

s. Database

t. Deployment

u. Unit test cases

Introduction:

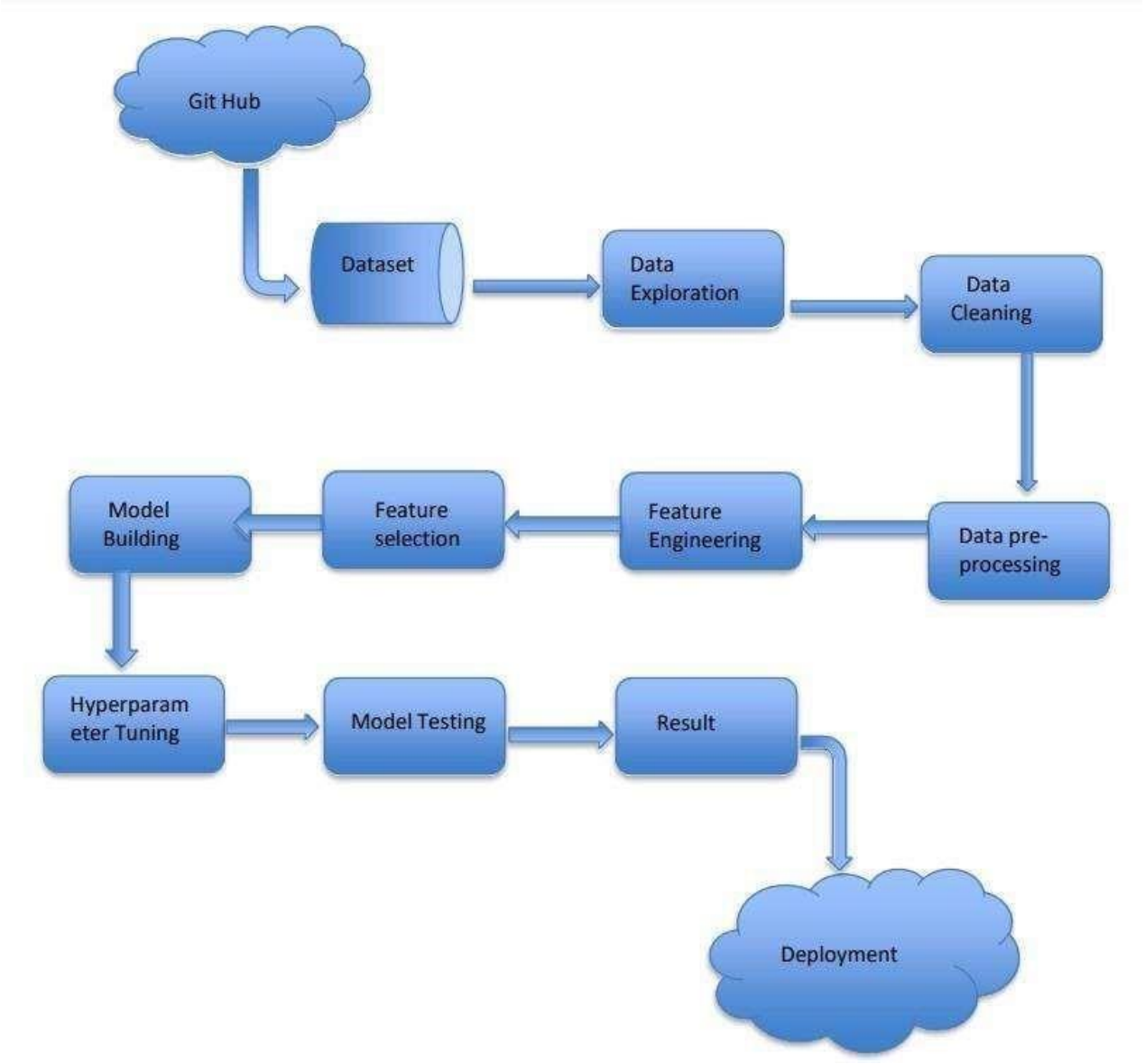
1.1. What is a Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for the Credit Card Default prediction code. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

1.1.1 Scope:

Low-level design (LLD) is a component-level design process that follows a step-bystep refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organisation may be defined during requirement analysis and then refined during data design work

Architecture:



Architecture Description:

1.2. . Data Description:

The dataset of this research has been published in Kaggle. Data Source:

https://www.kaggle.com/datasets/uciml/defaul t-of-credit-card-clients-dataset

The dataset is highly imbalanced which should be addressed for accurate predictions by the model; this dataset contains 25 attributes with 30,000 observations for the training and testing sets.

1.3. . Data Transformation:

In the Transformation Process, we will convert our original dataset which is in CSV Format to a pandas data frame. After reading the dataset we have two pandas data frame train and test concatenated into one data frame.

1.4. . Exploratory Data Analysis:

Exploratory Data Analysis, or EDA, is an important step in any Data Analysis or Data Science project EDt. A is the process of investigating the dataset to discover patterns, and anomalies (outliers), and form hypotheses based on our understanding of the dataset.

1.5. Data Insertion into Database:

a. Database Creation and connection - Create a database with name passed. If the database is already created, open the connection to the database. b. Table creation in the database.

c. Insertion of files in the table.

1.6. . Export Data from Database:

The data in a stored database is exported as a CSV file to be used for Data Preprocessing and Model Training.

3.6. Data Cleaning:

Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

3.7 Data Pre-processing:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always the case that we come across clean and formatted data.

3.8 Feature Engineering:

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process.

3.9 Feature Selection:

A feature selection algorithm can be seen as the combination of a search technique for proposing new feature subsets, along with an evaluation measure which scores the different feature subsets. The simplest algorithm is to test each possible subset of features finding the one which minimizes the error rate. This is an exhaustive search of the space and is computationally intractable for all but the smallest of feature sets.

3.10 Model Building:

A machine learning model is built by learning and generalizing from training data, then applying that acquired knowledge to new data it has never seen before to make predictions and fulfil its purpose. Lack of data will prevent you from building the model, and access to data isn't enough.

3.11 Random Forest:

A random forest classifier, a random forest is a Meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

3.12 Decision Tree Classifier:

Decision trees are the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

3.13 Logistic Regression:

Logistic regression is a statistical model used for binary classification. It predicts the probability of an event belonging to one of two classes. It uses the logistic function (sigmoid function) to transform the input variables into a value between 0 and 1, representing the estimated probability. The model provides interpretable coefficients for each input variable, indicating their influence on the predicted probability. Logistic regression is trained using maximum likelihood estimation and can be evaluated using metrics such as accuracy.

3.14 Support Vector Classification:

The Support Vector Classifier (SVC), also known as Support Vector Machine (SVM), is a classification algorithm. It finds an optimal hyperplane that separates data points of different classes with the maximum margin. SVC can handle nonlinear data by using the kernel trick, and it aims to find a hyperplane with the largest margin between support vectors. It is trained by solving an optimization problem, and it can classify new data points based on their position relative to the hyperplane.

3.15 K Nearest Neighbors Classification:

K Nearest Neighbors (KNN) Classification is a simple machine learning algorithm used for classification tasks. It classifies a new data point based on the majority vote of its K nearest neighbors in the feature space. The algorithm measures the distance between the new point and the existing labeled data points, and assigns the class label that is most common among the K nearest neighbors. KNN is a non-parametric algorithm, meaning it doesn't make any assumptions about the underlying data distribution. It is easy to understand and implement, but can be sensitive to the choice of K and the distance metric used for measuring proximity.

3.16. Boosting:

Boosting is a machine learning ensemble technique where multiple weak learners are combined to create a strong learner. It works by sequentially training models, where each subsequent model corrects the errors of its predecessors. In essence, it focuses more on the data points that previous models struggled with, thereby improving overall accuracy. Common algorithms for boosting include AdaBoost, Gradient Boosting.

3.17. Model Validation:

For machine learning systems, we should be running model evaluation and model tests in parallel. Model evaluation covers metrics and plots which summarise performance on a validation or test dataset. Model testing involves explicit checks for behaviours that we expect our model to follow.

3.18 Flask:

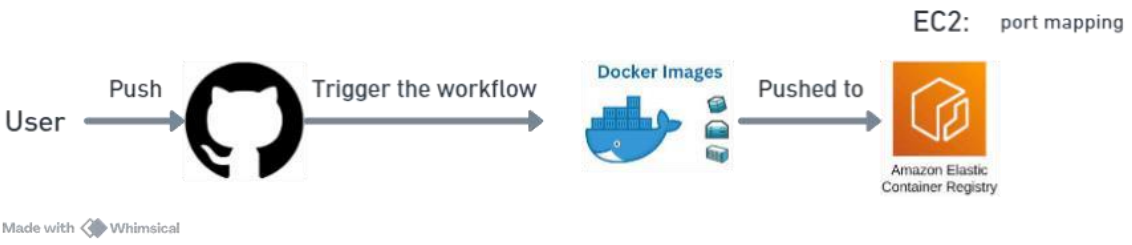
Flask is a micro web framework written in python. Here we use flask to create an API that allows us to send data, and receive a prediction as a response. Flask supports extensions that can add application features as if they were implemented in Flask itself.

4. Database:

Cassandra DB is used to retrieve, insert, delete and update the customer input database. Flask is a micro web framework written in python. Here we use the flask to create an API that allows sending data, and receives a prediction as a response.

Flask API's act as an interface between the user, ML- Model and DB. Users interact with the model and the result will be returned to the user by API. The Flask API will collect all the customer input data stored in Cassandra DB.

5.Deployment:



Unit test Cases:

Test Case Description	Pre-requisite	Expected Result
Verify the user should able to see their input data	1. Application is accessible 2. Application is responsive	Users can see their data in the form.
Verify the user can edit their information in the form	1.Application is responsive	User should be able to edit all input field
Verify whether the user gets Submit button to submit the inputs.	1.Application is accessible 2.Application is responsive	User should able to submit values
Verify whether the wrong information should not be submitted by the user	Application is secured	Correct information should be submitted